CPSC 449: Assignment 3

Fall 2014

Due: Monday, October 27, 2014 at **12:00 PM** noon

1. [40%] A *polynomial in one variable* (or "polynomial" in short) is defined inductively as follows:

   - a **`Int`** constant is a polynomial,
   - *the* variable is a polynomial (**Hint:** every single-variable polynomial has exactly one variable, and how it is named is **NOT** important),
   - if P and Q are polynomials, then $P + Q$ is a polynomial, and
   - if P and Q are polynomials, then $P \times Q$ is a polynomial.

   Except for the above there is no other polynomial.

   (a) [10%] Define an algebraic type **`Poly`** to represent polynomials.

   (b) [5%] Give a Haskell expression that constructs a **`Poly`** representation of the polynomial "$1 + x + x^3$."

   (c) [5%] Define a function **`showPoly :: Poly -> String`** such that **`(showPoly p)`** returns a string representation of the polynomial **`p`**. You need this function for debugging the following part. **Hint:** There is no need to do factorization or simplification.

   (d) [10%] Define a function **`derivative :: Poly -> Poly`** such that **`(derivative p)`** returns the first derivative of **`p`**. **Hint:** It is time to pull out your first-year calculus textbook.

   (e) [10%] State the "Principle of Structural Induction for Polynomial".

2. [20%] **[Thompson]** exercise 14.44. **Hint:** The definition of **`depth`** is given on page 334 of **[Thompson]**.

3. [20%] Use **`map`**, **`filter`**, and/or **`foldr`** to implement the following Haskell functions.

   (a) [10%] **`and :: [Bool] -> Bool`**, where **`(and xs)`** returns **`True`** iff no member of **`xs`** is **`False`**.

   (b) [10%] **`count :: (Integer->Bool) -> [Integer] -> Integer`**, where **`(count p xs)`** returns the number of members of **`xs`** that satisfy the predicate **`p`**.

4. (a) [14%] **[Thompson]** exercise 10.9.

   (b) [6%] **[Thompson]** exercise 10.10.